

2

Japanese Patent Laid-open Publication No.: 07-121379 A

Publication date : May 12, 1995

Applicant : NEC Software Ltd..

Title : Multiple Languages Mixed Compiler

5

(57) [Abstract]

[Object] To make translation of a mixed multiple languages source program described by multiple languages possible using a single compiler and improve translation speed.

10 [Configuration] A reading unit 3, when reading a multiple languages source program by a language A and a language B, separates a language A description part and a language B description part by a language B description part recognition unit 4, and intermediate texts are
15 generated for each language by intermediate text generating units. In a syntax analysis unit 7, each intermediate text is syntax analyzed and translated based on the language format of each text by an independent syntax analysis and translating unit for each language and a translation result
20 for each language is generated. These translation results are combined in an object generating unit 10 by a translation results combining unit 11 and a compile unit 13 is generated by an object generating unit 13.

2

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開平7-121379

(43) 公開日 平成7年(1995)5月12日

(51) IntCl.⁸
G 0 6 F 9/45

識別記号

片内整理番号

F I

技術表示箇所

9292-5B

G 0 6 F 9/44

3 2 2 L

審査請求 未請求 請求項の数2 O L (全9頁)

(21) 出願番号 特願平5-269329

(22) 出願日 平成5年(1993)10月28日

(71) 出願人 000232092

日本電気ソフトウェア株式会社

東京都江東区新木場一丁目18番6号

(72) 発明者 俣野 圭吾

東京都港区高輪二丁目17番11号 日本電気
ソフトウェア株式会社内

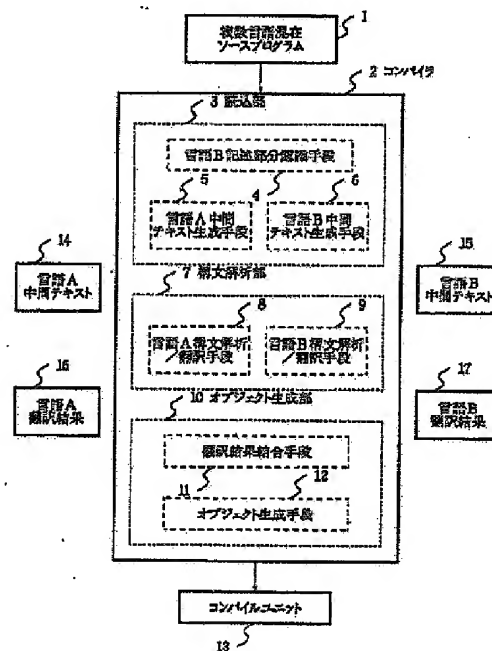
(74) 代理人 弁理士 京本 直樹 (外2名)

(54) 【発明の名称】 複数言語混在コンパイラ

(57) 【要約】

【目的】 複数の言語により記述されている、複数言語混在のソースプログラムを単一のコンパイラにより翻訳できるようにし、翻訳速度の向上をはかる。

【構成】 読込部3が、言語A、言語Bにより記述された複数言語混在のソースプログラムを読み込む際に、言語B記述部分認識手段4により、言語A記述部分と、言語B記述部分を切りわけ、中間テキスト生成手段により、各言語ごとに別々の中間テキストを生成する。それぞれの中間テキストは、構文解析部7において、各言語ごとに独立した構文解析翻訳手段により、それぞれの言語仕様に則った構文解析と翻訳が行われ、各言語ごとの翻訳結果を生成する。これらの翻訳結果は、オブジェクト生成部10において、翻訳結果結合手段11により結合され、オブジェクト生成手段12によってコンパイルユニット13を生成する。



【特許請求の範囲】

【請求項 1】 電子計算機のプログラム言語において、複数の言語仕様の異なる言語により記述されているソースプログラム中の、言語が切り換わる切れ目を認識する認識手段と、

ソースプログラムを、言語の切れ目によって切りわけ、言語ごとに中間テキストを生成する中間テキスト生成手段と、

それぞれの中間テキストを、言語ごとに構文解析および翻訳する構文解析翻訳手段と、

翻訳された複数の翻訳結果を結合する翻訳結果結合手段と、

結合された中間テキストをオブジェクトコードに変換するオブジェクト生成手段とを含むことを特徴とする複数言語混在コンパイラ。

【請求項 2】 読込部で、第 1 のソースプログラムから 1 レコード入力し、ソースの終了でなければ、そのレコード中に第 1 の言語による記述の開始キーワードの有無を判定し、開始キーワードが存在しない場合、第 2 の言語の言語仕様に基づいて第 2 の言語中間テキストを作成し、開始キーワードが存在する場合、それ以降のソースから前記第 1 の言語の終了キーワードを捜し出し、前記第 1 の言語の言語仕様に基づいて第 1 の言語中間テキストを作成し、前記第 2 の言語中間テキスト上の、前記第 1 の言語テキストが存在した部位に、前記第 1 の言語中間テキスト結合情報を埋め込み、前記読込部で作成された前記第 1 の言語中間テキスト、前記第 2 の言語中間テキストを、構文解析部で読み込み構文解析および翻訳した後、第 1、第 2 の言語翻訳結果として出力し、前記構文解析部で、前記第 2 の言語中間テキストを入力し、中間テキストの終了でなければ、入力されたテキストが前記読込部で埋め込まれた前記第 1 の言語の埋め込み情報であるかを調べ、前記第 1 の言語の埋め込み情報でなければ、第 2 の言語構文解析翻訳手段により、構文解析と翻訳を行い、第 2 の言語翻訳結果として出力し、中間テキストの内容が前記第 1 の言語の埋め込み情報の場合は、その埋め込み情報に対応する前記第 1 の言語中間テキストを入力し、第 1 の言語構文解析翻訳手段により、構文解析と翻訳を行い、前記第 1 の言語翻訳結果として出力し、このとき、前記第 1 の言語の構文解析時のエラーについての情報を、前記第 2 の言語翻訳結果上の、対応する言語の記述があった位置に埋め込み、前記構文解析部で作成された第 1、第 2 の言語翻訳結果を、オブジェクト生成部で読み込み、コンパイルユニットを作成し、オブジェクト生成部で、前記第 2 の言語翻訳結果を入力し、翻訳結果の終了でなければ、入力された情報が前記第 1 の言語の埋め込み情報かを調べ、前記第 1 の言語の埋め込み情報ならば、それに対応する第 1 の言語の翻訳結果を結合し、この結合された翻訳結果に対してオブジェクトを生成することを特徴とする複数言語混在コ

ンパイル方法。

【発明の詳細な説明】

【0001】

【産業上の利用分野】本発明は、電子計算機プログラム言語の処理プロセッサに関し、特に、複数言語混在コンパイラにする。

【0002】

【従来の技術】COBOL プログラム埋め込み型データベース言語 SQL のように、それぞれ異なった言語仕様を持つ複数のプログラム言語を用いて一つのソースプログラムを記述した場合、複数言語が混在したソースプログラムが生成される。

【0003】このようなソースプログラムをコンパイルする場合、従来はプリコンパイラによって、埋め込まれた言語によるソースプログラムの記述をもう一方の言語による記述に置換した後、その言語のコンパイラを起動することにより実現している。

【0004】この例を、図面により説明する。図 5 は、従来のプリコンパイラ方式の 1 例である。複数言語が混在したソースプログラム 500 は、言語 B プリコンパイラ 501 によって、言語 A のみで記述されたソースプログラム 507 へと変換され、その後、言語 A コンパイラ 508 によりコンパイルユニット 512 へと変換されている。

【0005】

【発明が解決しようとする課題】上述した従来の技術では、プリコンパイラの起動後に、もう一度コンパイラを起動するため、コンパイルユニット生成までの時間が余分にかかるという問題点がある。

【0006】また、一方の言語による記述を、もう一方の言語による記述に一旦置き換えるため、一方の言語仕様が、もう一方の言語仕様により制限されることがある。

【0007】また、ソースの記述言語の種類を増やすことが困難である。

【0008】

【課題を解決するための手段】本発明の複数言語混在コンパイラは、電子計算機のプログラム言語において、複数の言語仕様の異なる言語により記述されているソースプログラム中の、言語が切り換わる切れ目を認識する認識手段と、ソースプログラムを、言語の切れ目によって切りわけ、言語ごとに中間テキストを生成する中間テキスト生成手段と、それぞれの中間テキストを、言語ごとに構文解析および翻訳する構文解析翻訳手段と、翻訳された複数の翻訳結果を結合する翻訳結果結合手段と、結合された中間テキストをオブジェクトコードに変換するオブジェクト生成手段とを含むことを特徴とする。

【0009】

【実施例】次に、本発明について図面を参照して説明する。

【0010】図 1 は、本発明の一実施例を示す構成図で

ある。本発明の一実施例である複数言語混在コンパイラ2は、複数言語混在ソースプログラム1を入力し、コンパイルユニット13を出力する。

【0011】言語混在プロセッサ2は、言語B記述部分認識手段4、言語A中間テキスト生成手段5、言語B中間テキスト生成手段6よりなる読込部3と、言語A構文解析翻訳手段8、言語B構文解析翻訳手段9よりなる構文解析部7と、翻訳結果結合手段11、オブジェクト生成手段12よりなるオブジェクト生成部10とにより構成される。

【0012】図2は、読込部3の動作を表した流れ図である。

【0013】図3は、構文解析部7の動作を表した流れ図である。

【0014】図4は、オブジェクト生成部10の動作を表した流れ図である。

【0015】以下に、図1から図4までを用いて、図1に示す実施例の動作を説明する。

【0016】言語A、言語Bによって書かれた複数言語混在ソースプログラム1は、読込部3により読み込まれ、言語A中間テキスト14、言語B中間テキスト15を生成する。

【0017】読込部3は、ソースプログラム1から1レコード入力（ステップ201）し、ソースの終了（ステップ202）でなければ、そのレコード中に言語Bによる記述の開始キーワードの有無を判定（ステップ203）する。

【0018】開始キーワードが存在しない場合、そのレコードは言語Aにより記述されている為、言語Aの言語仕様に基づいて言語A中間テキスト14を作成（ステップ204）する。

【0019】開始キーワードが存在する場合、それ以降のソースから言語B終了キーワードを捜し出し（ステップ205）、開始キーワードから終了キーワードまでのソースは言語Bにより記述されているので、言語Bの言語仕様に基づいて言語B中間テキスト15を作成（ステップ206）する。

【0020】そして、言語A中間テキスト14上の、言語Bテキストが存在した部位に、言語B中間テキスト結合情報を埋め込む（ステップ207）。

【0021】読込部3で作成された、言語A中間テキスト14、言語B中間テキスト15は、構文解析部7で読み込まれ構文解析および翻訳された後、言語A翻訳結果16、言語B翻訳結果17に出力される。

【0022】構文解析部7は、言語A中間テキスト14を入力（ステップ301）し、中間テキストの終了（ステップ302）でなければ、入力されたテキストが読込部3で埋め込まれた言語B埋め込み情報であるかを調べ（ステップ303）、言語B埋め込み情報でなければ、言語A構文解析／翻訳手段8により、構文解析と翻訳を

行い（ステップ304）、言語A翻訳結果16を出力する。

【0023】中間テキストの内容が言語B埋め込み情報の場合は、その埋め込み情報に対応する言語B中間テキスト15を入力（ステップ305）し、言語B構文解析／翻訳手段9により、構文解析と翻訳を行い（ステップ306）、言語B翻訳結果17を出力する。このとき、言語Bの構文解析時のエラーについての情報は、言語A翻訳結果16上の、対応する言語Bの記述があった位置に埋め込まれる（ステップ307）。

【0024】構文解析部7で作成された、言語A翻訳結果16、言語B翻訳結果17は、オブジェクト生成部10で読み込まれ、コンパイルユニット13を作成する。

【0025】オブジェクト生成部10は、言語A翻訳結果16を入力し（ステップ401）、翻訳結果の終了（ステップ402）でなければ、入力された情報が言語B埋め込み情報かを調べ（ステップ403）、言語B埋め込み情報ならば、それに対応する言語B翻訳結果17を結合する（ステップ405）。そして、この結合された翻訳結果に対してオブジェクトを生成する（ステップ404）。

【0026】

【発明の効果】本発明は、以上に説明したように、複数の言語により記述されているソースプログラムを翻訳する場合、一方の言語による記述をもう一方の言語による記述に置き換えるという処理を行わず、単一のコンパイラによって処理を行うために、翻訳速度が向上するという効果を奏する。

【0027】また、それぞれの言語の構文解析、翻訳が独立して行われるので、一方の言語の記述が、他の言語の言語仕様により制限を受けることがない。

【図面の簡単な説明】

【図1】本発明の一実施例の構成図である。

【図2】図1中の読込部の動作を表した流れ図である。

【図3】図1中の構文解析部の動作を表した流れ図である。

【図4】図1中のオブジェクト生成部の動作を表した流れ図である。

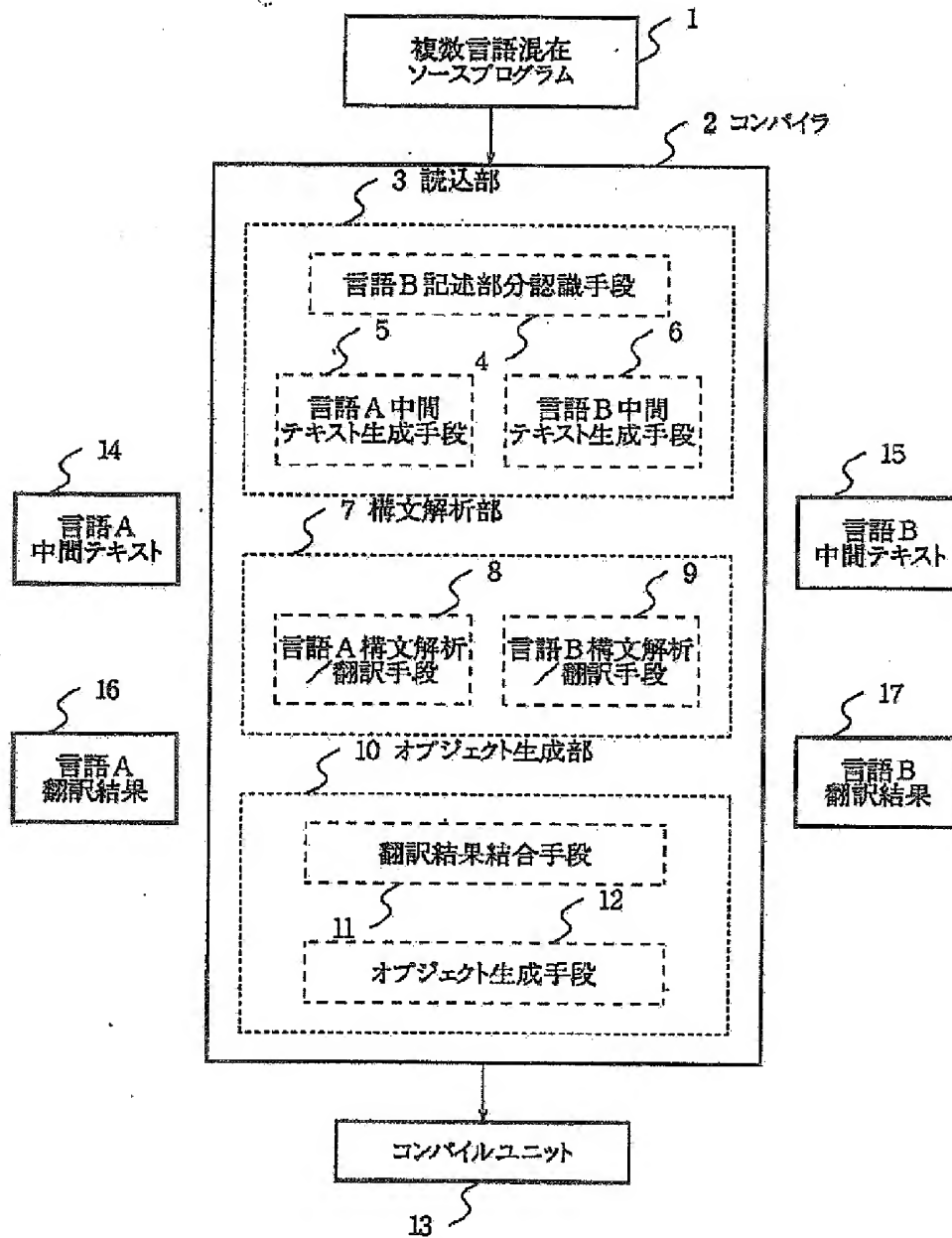
【図5】従来のプリコンパイラ方式を例示する構成図である。

【符号の説明】

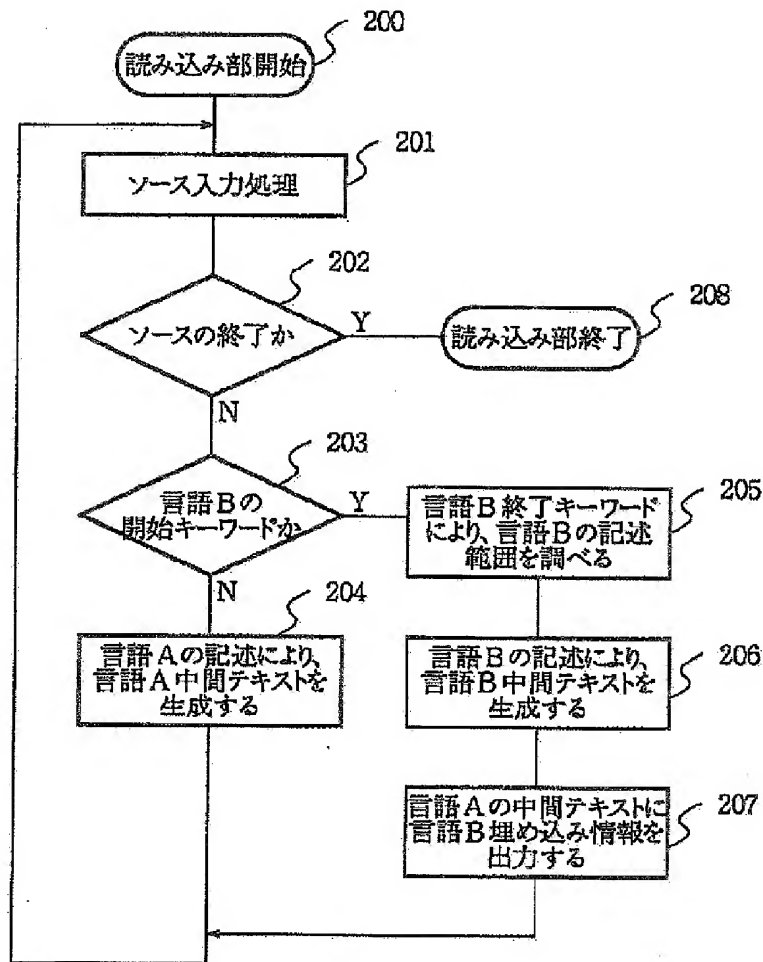
- 1 言語A、言語Bが混在しているソースプログラム
- 2 複数言語混在コンパイラ
- 3 読込部
- 4 言語B記述部分認識手段
- 5 言語A中間テキスト生成手段
- 6 言語B中間テキスト生成手段
- 7 構文解析部
- 8 言語A構文解析／翻訳手段
- 9 言語B構文解析／翻訳手段

5	<p>10 オブジェクト生成部</p> <p>11 翻訳結果結合手段</p> <p>12 オブジェクト生成手段</p> <p>13 コパイルユニット</p> <p>14 言語A中間テキスト生成手段が生成した言語A中間テキスト</p> <p>15 言語B中間テキスト生成手段が生成した言語B中間テキスト</p> <p>16 言語A構文解析翻訳手段が生成した言語A構文解析、翻訳結果</p> <p>17 言語B構文解析翻訳手段が生成した言語B構文解析、翻訳結果</p> <p>200 読込部開始</p> <p>201 ソース入力処理</p> <p>202 ソースの終わり判定処理</p> <p>203 言語Bの記述部分の先頭の判定処理</p> <p>204 言語A記述による言語Aの中間テキストの生成処理</p> <p>205 言語Bの記述部分の終端を判定し、言語Bの記述の取得処理</p> <p>206 言語B記述による言語Bの中間テキストの生成処理</p> <p>207 言語B埋め込み情報の言語Aの中間テキスト上への生成処理</p> <p>208 読込部終了</p> <p>300 構文解析部開始</p> <p>301 言語A中間テキスト入力処理</p> <p>302 言語A中間テキストの終わり判定処理</p> <p>303 言語B埋め込み情報判定処理</p> <p>304 言語A構文解析／翻訳処理</p> <p>305 言語B埋め込み情報に対応する言語B中間テキスト入力処理</p> <p>306 言語B構文解析、翻訳処理</p>	6	<p>307 言語B構文解析翻訳情報の言語A翻訳結果上への生成処理</p> <p>308 構文解析部終了</p> <p>400 オブジェクト生成部開始</p> <p>401 言語A翻訳結果入力処理</p> <p>402 言語A翻訳結果の終わり判定処理</p> <p>403 言語B埋め込み情報判定処理</p> <p>404 オブジェクト生成処理</p> <p>405 言語B埋め込み情報に対応する言語B翻訳結果</p> <p>10 結合処理</p> <p>406 オブジェクト生成部終了</p> <p>500 言語A、言語Bが混在しているソースプログラム</p> <p>501 言語Bプリコンパイラ</p> <p>502 言語B記述部分認識手段</p> <p>503 言語B中間テキスト生成手段</p> <p>504 言語B構文解析／翻訳手段</p> <p>505 言語Bオブジェクト生成手段</p> <p>506 オブジェクト結合手段</p> <p>20 507 言語Aによるソースプログラム</p> <p>508 言語Aコンパイラ</p> <p>509 言語A中間テキスト生成手段</p> <p>510 言語A構文解析翻訳手段</p> <p>511 言語Aオブジェクト生成手段</p> <p>512 コンパイルユニット</p> <p>513 ソースプログラム中の言語A記述部分のテキスト</p> <p>514 言語B中間テキスト</p> <p>515 言語B翻訳結果</p> <p>30 516 言語Aで記述された言語Bプリコンパイラのオブジェクト</p> <p>517 言語A中間テキスト</p> <p>518 言語A翻訳結果</p>
---	---	---	--

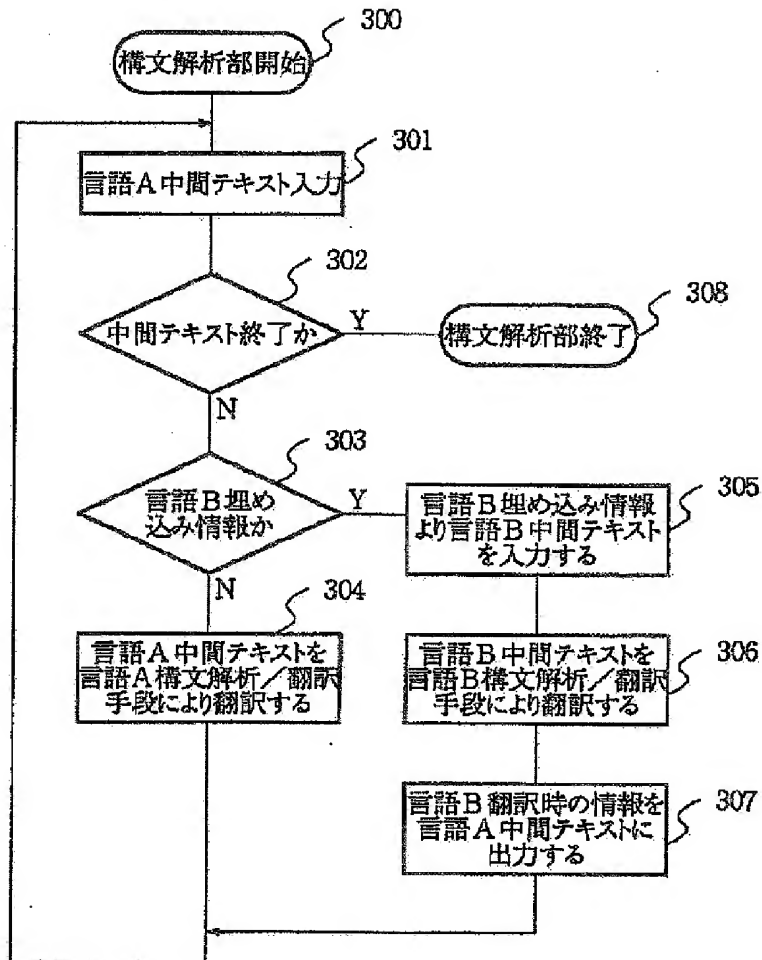
【図1】



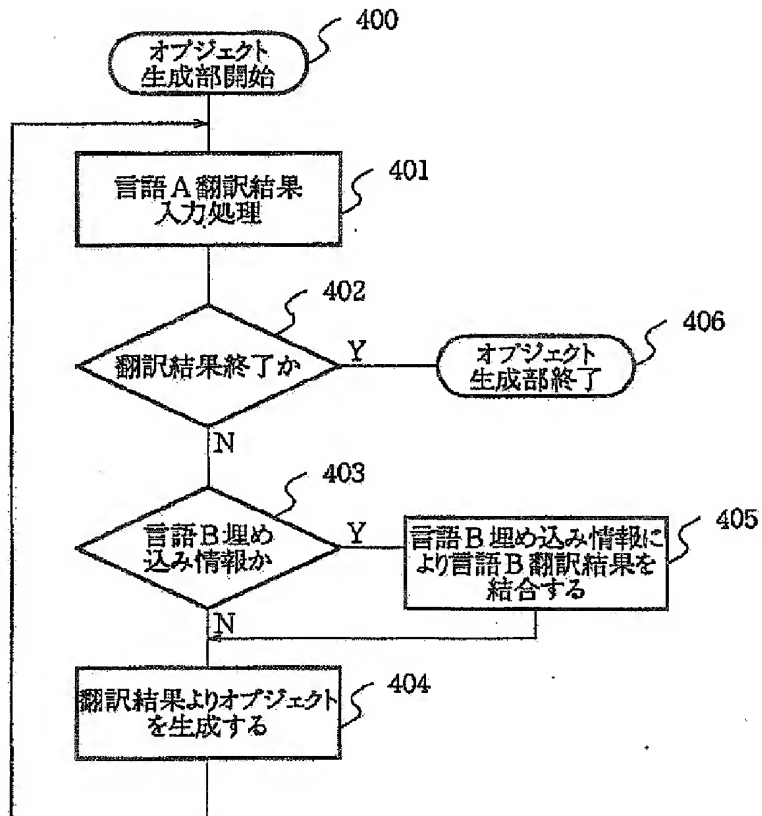
【図 2】



【図3】



【図4】



【図5】

